Thresh 🌾: A Unified, Customizable and Deployable Platform for Fine-Grained Text Evaluation

David Heineman, Yao Dou, Wei Xu
School of Interactive Computing, Georgia Institute of Technology
{david.heineman, douy}@gatech.edu; wei.xu@cc.gatech.edu

Abstract

Fine-grained, span-level human evaluation has emerged as a reliable and robust method for evaluating text generation tasks such as summarization, simplification, machine translation and news generation, and the derived annotations have been useful for training automatic metrics and improving language models. However, existing annotation tools lack adaptability to be extended to different domains or languages, or modify annotation settings according to user needs. In this paper, we introduce Thresh 🌾, a unified, customizable and deployable platform for fine-grained evaluation. With a single YAML configuration file, users can build and test an annotation interface for any framework within minutes – all in one web browser window. Thresh is publicly accessible at https://thresh.tools. **This work has appeared previously at the EMNLP 2023 System Demonstrations.**

1 Introduction

We present Thresh 🌾: a unified and customizable platform for building, distributing and orchestrating fine-grained human evaluation for text generation in an efficient and easy-to-use manner. Our platform allows users to create, test and deploy an evaluation framework within minutes, all in a single browser window and has already been used to orchestrate large-scale data annotation (Heineman et al., 2023). Thresh also serves as a *community hub* for fine-grained evaluation frameworks and annotation data, all presented in a unified format. The following are the design principles of Thresh:

- **Unified:** Thresh standardizes fine-grained evaluation into two key components: span selection and span annotation. Users can easily implement any framework by writing a YAML template file (see Figure 1), and Thresh will build the corresponding annotation interface. All resulting annotations adhere to a consistent JSON format.
- **Customizable:** Thresh offers extensive customization to meet a wide range of user needs. This includes different span selection methods from subword to word-level, diverse annotation options including custom questions and text boxes to handle arbitrary typologies, as well as customized interface elements in any language.
- **Deployable:** Thresh supports a range of deployment options for annotation projects of various scales. Small-scale linguistic inspections (e.g., manual ablation studies) can be directly hosted on the platform. For larger projects, users can host their template in a GitHub repository and connect to Thresh. Thresh is also compatible with crowdsourcing platforms such as Prolific[1] and Amazon MTurk[2].
- **Contributive:** Thresh also operates as a community hub where users can contribute and access a wide variety of fine-grained evaluation frameworks and their annotation data. Currently, it includes 11 frameworks as displayed in Table 1.
- **End-to-End:** Beyond facilitating the creation and deployment of evaluation frameworks, Thresh streamlines every step of the annotation process. It offers functions for authors to publish their typologies as research artifacts and a supplementary Python library, released under the Apache 2.0 license, to help data collection.[3]

2 Fine-Grained Text Evaluation

Thresh formulates fine-grained text evaluation as two components: *span selection* and *span annotation*. During development, users define their annotation typology and interface features using a YAML template (see Sec 3 and Fig 1 for more details). Based on the configuration, Thresh then

[1] https://www.prolific.co
[2] https://www.mturk.com
[3] https://www.pypi.org/project/thresh

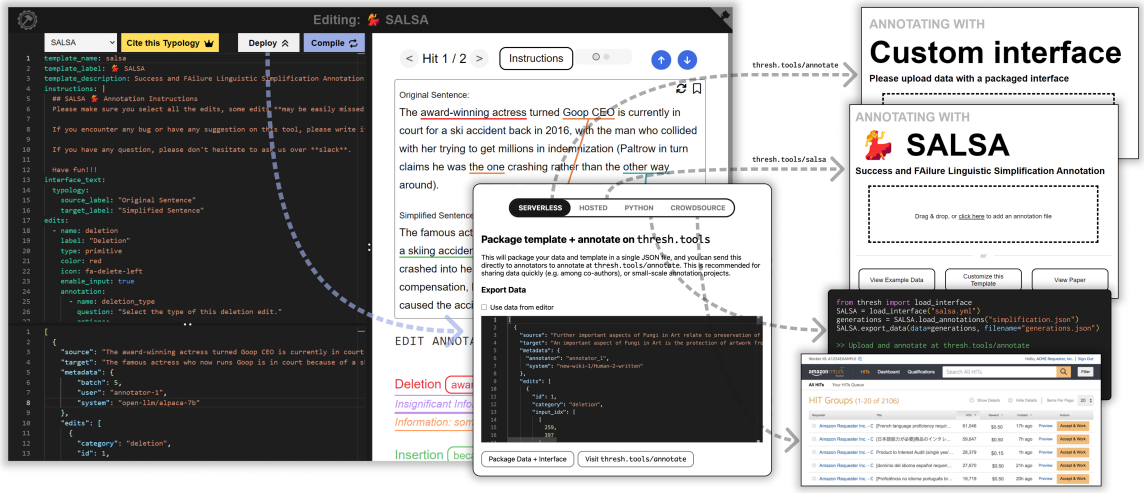

Figure 1: Thresh deployment workflow. Users build and test their template and then deploy with one of 4 options.

constructs an annotation interface that integrates both components.

2.1 Span Selection

Each annotation instance consists of the *source*, *target* and *context*. For example, in open-ended text generation (Zellers et al., 2019), the source is a starting sentence and the target is a model-generated continuation. In text simplification (Xu et al., 2016), the source would be a complex sentence or paragraph, and the target would be the generated simplification. The context holds additional relevant information, such as a prompt instruction, a retrieved Wikipedia page, or a dialogue history. During the span selection stage, annotators select relevant spans, referred to as *Edits*, in the source and target, following the edit category definitions outlined in the typology.

2.2 Span Annotation

In the YAML file, users define the typology in a decision tree structure to further categorize the selected spans into fine-grained types. Unlike previous work which presents all fine-grained edit types to annotators simultaneously, Thresh recursively compiles the annotation interface. Annotators thus will answer a series of questions or follow-up questions under each edit type. This tree structure enables support for complex error typologies. Thresh supports binary, three and five-scale questions with customized label names, as well as text boxes for tasks that require human post-editing or explanations. With these features, our interface supports complex annotation schemes in a flexible and easily extensible way.

3 Interactive Interface Builder

To alleviate the time consuming process of customizing and hosting front-end code — even building custom databases in some cases — Thresh implements an in-browser interface builder, which allows users to create, test and deploy a fine-grained interface within a single web browser page, as depicted in Figure 1. Users write a YAML template to construct their interface and provide data with a JSON file. The *Compile* button allows users to preview their interface, and the *Deploy* button presents instructions for different deployment options.

Template Hub. As Thresh aims to facilitate easy use and distribution of fine-grained evaluation frameworks, it provides a template hub that makes it simple for any NLP practitioner to access a framework with their own data. Alongside the 10 tutorial templates that explain each interface feature, the annotation builder currently includes 11 widely used inspection and evaluation typologies across major text generation tasks. Table 1 (on Page 2) lists each framework, its associated task and link to our implementation.

Unified Data Model. To ensure compatibility, we create conversion scripts that adapt these annotations to our unified format. Our scripts are designed to be *bidirectional*, meaning data published for these typologies can be converted to our format and back without data loss. Our unified fine-grained data format allows smooth transfer of analysis, agreement calculation and modeling code between different projects. We believe this will support research in learning with multi-task fine-grained training setups or model feedback.

Ethical Considerations

We do not anticipate any ethical issues pertaining to the topics of fine-grained evaluation supported by our interface. Nevertheless, as Thresh lowers the barrier to fine-grained evaluation, vast ethical responsibility falls upon practitioners using our platform to prevent the exploitation of crowdsource workers, through fair pay (Fort et al., 2011) and safeguards against exposure to harmful or unethical content (Shmueli et al., 2021). As task difficulty and complexity scales with the granularity of data collected, increasing care must be taken for training annotators adequately and to scale pay accordingly (Williams et al., 2019).

Acknowledgments

This research is supported in part by the NSF awards IIS-2144493 and IIS-2112633, ODNI and IARPA via the HIATUS program (contract 2022-22072200004). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of NSF, ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeño, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. 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 5636–5646, Hong Kong, China. Association for Computational Linguistics.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A. Smith, and Yejin Choi. 2022a. Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7250–7274, Dublin, Ireland. Association for Computational Linguistics.

Yao Dou, Chao Jiang, and Wei Xu. 2022b. Improving large-scale paraphrase acquisition and generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9301–9323, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Karën Fort, Gilles Adda, and K. Bretonnel Cohen. 2011. Last words: Amazon Mechanical Turk: Gold mine or coal mine? *Computational Linguistics*, 37(2):413–420.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. SNaC: Coherence error detection for narrative summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 444–463, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

David Heineman, Yao Dou, Mounica Maddela, and Wei Xu. 2023. Dancing between success and failure: Edit-level simplification evaluation using SALSA. *arXiv preprint arXiv:2305.14458*.

Marcelo Yuji Himoro and Antonio Pareja-Lora. 2020. Towards a spell checker for Zamboanga Chavacano orthography. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2685–2697, Marseille, France. European Language Resources Association.

Chao Jiang, Wei Xu, and Samuel Stevens. 2022. arXivEdits: Understanding the human revision process in scientific writing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9420–9435, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Boaz Shmueli, Jan Fell, Soumya Ray, and Lun-Wei Ku. 2021. Beyond fair pay: Ethical implications of NLP crowdsourcing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3758–3769, Online. Association for Computational Linguistics.

Alex C Williams, Gloria Mark, Kristy Milland, Edward Lank, and Edith Law. 2019. The perpetual work

life of crowdworkers: How tooling practices increase fragmentation in crowdwork. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–28.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. *arXiv preprint arXiv:2306.01693*.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Framework	Task	Released	Link
Evaluation			
MQM (Freitag et al., 2021)	Translation	✓	thresh.tools/mqm
FRANK (Pagnoni et al., 2021)	Summarization	✓	thresh.tools/frank
SNaC (Goyal et al., 2022)	Narrative Summarization	✓	thresh.tools/snac
Scarecrow (Dou et al., 2022a)	Open-ended Generation	✓	thresh.tools/scarecrow
SALSA (Heineman et al., 2023)	Simplification	✓	thresh.tools/salsa
ERRANT (Bryant et al., 2017)	Grammar Error Correction	✗	thresh.tools/errant
FG-RLHF (Wu et al., 2023)	Fine-Grained RLHF	✓	thresh.tools/fg-rlhf
Inspection			
MultiPIT (Dou et al., 2022b)	Paraphrase Generation	✗	thresh.tools/multipit
CWZCC (Himoro and Pareja-Lora, 2020)	Zamboanga Chavacano Spell Checking	✗	thresh.tools/cwzcc
Propaganda (Da San Martino et al., 2019)	Propaganda Analysis	✓	thresh.tools/propaganda
arXivEdits (Jiang et al., 2022)	Scientific Text Revision	✓	thresh.tools/arxivedits

Table 1: Existing typologies implemented on Thresh with their associated link. *Released* indicates whether the annotated data is released.