Demonstrating PAR4SEM - A Semantic Writing Aid with Adaptive Paraphrasing

Seid Muhie Yimam **Chris Biemann**
Language Technology Group
Department of Informatics
Universität Hamburg, Germany
{yimam,biemann}@informatik.uni-hamburg.de

Abstract

In this paper, we present PAR4SEM, a semantic writing aid tool based on adaptive paraphrasing. Unlike many annotation tools that are primarily used to collect training examples, PAR4SEM is integrated into a real word application, in this case a writing aid tool, in order to collect training examples from usage data. PAR4SEM is a tool, which supports an adaptive, iterative, and interactive process where the underlying machine learning models are updated for each iteration using new training examples from usage data. After motivating the use of ever-learning tools in NLP applications, we evaluate PAR4SEM by adopting it to a text simplification task through mere usage.

1 Introduction

Natural language processing and semantic applications that depend on a machine learning component require training data, i.e. examples from which the machine learning algorithm learns from. The training datasets require, most of the time, manual annotation. Usually, such annotations are conducted in a predefined cycle of annotation activities. Once the annotation problem is identified, a standalone annotation tool along with the annotation guideline is developed. At the end of the annotation cycle, the collected dataset is fed to the machine learning component, which produces a static model that can be used thereafter in an application.

Possible limitations of these annotation approaches are: 1) Developing a standalone annotation tool is costly, sometimes expert or specially trained annotators are required. 2) There is no direct way to evaluate the dataset towards its effectiveness for the real-world application. 3) It suffers from what is known as *concept drift*, as the annotation process is detached from the target application, the dataset might get obsolete over time.

In this regard, we have dealt specifically with the semantic annotation problem, using an adaptive, integrated, and personalized annotation process. By *adaptive*, we mean that target applications do not require pre-existing training data, rather it depends on the usage data from the user. The machine learning model then adapts towards the actual goal of the application over time. Instead of developing a standalone annotation tool, the collection of training examples is *integrated* inside a real-world application. Furthermore, our approach is *personalized* in a sense that the training examples being collected are directly related to the need of the user for the application at hand. After all, the question is not: how good is the system *today*? It is rather: how good will it be *tomorrow* after we use it today?

Thus, such adaptive approaches have the following benefits:

Suggestion and correction options: Since the model immediately starts learning from the usage data, it can start predicting and suggesting recommendations to the user immediately. Users can evaluate and correct suggestions that in turn help the model to learn from these corrections.

Less costly: As the collection of the training data is based on usage data, it does not need a separate annotation cycle.

Personalized: It exactly fits the need of the target application, based on the requirement of the user.

Model-Life-Long Learning: As opposed to static models that once deployed on the basis of training data, adaptive models incorporate more training data the longer they are used, which should lead to better performance over time.

We have developed PAR4SEM, a semantic writing aid tool using an adaptive paraphrasing component, which is used to provide context-aware lexical paraphrases while composing texts. The tool incorporates two adaptive models, namely target identification and candidate ranking. The adaptive target identification component is a clas-

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (System Demonstrations), pages 48–53
Brussels, Belgium, October 31–November 4, 2018.

sification algorithm, which learns how to automatically identify target units (such as words, phrases or multi-word expressions), that need to be paraphrased. When the user highlights target words (*usage data*), it is considered as a training example for the adaptive model.

The adaptive ranking model is a learning-to-rank machine learning algorithm, which is used to re-rank candidate suggestions provided for the target unit. We rely on existing paraphrase resources such as PPDB 2.0, WordNet, distributional thesaurus and word embeddings (see Section 2.1.1) to generate candidate suggestions.

Some other examples for adaptive NLP setups include: 1) online learning for ranking, example Yogatama et al. (2014) who tackle the pairwise learning-to-ranking problem via a scalable online learning approach, 2) adaptive machine translation (MT), e.g. Denkowski et al. (2014) describe a framework for building adaptive MT systems that learn from post-editor feedback, and 3) incremental learning for spam filtering, e.g. Sheu et al. (2017) use a window-based technique to estimate for the condition of concept drift for each incoming new email.

We have evaluated our approach with a lexical simplification task use-case. The lexical simplification task contains complex word identification (*adaptive target identification*) and simpler candidate selection (*adaptive ranking*) components.

As far as our knowledge concerned, PAR4SEM is the first tool in the semantic and NLP research community, where adaptive technologies are integrated into a real-world application. PAR4SEM is open source[1] and the associated data collected for the lexical simplification use-case are publicly available. The live demo of PAR4SEM is available at `https://ltmaggie.informatik.uni-hamburg.de/par4sem`.

2 System Architecture of PAR4SEM

The PAR4SEM system consists of backend, frontend, and API components. The backend component is responsible for NLP related pre-processing, adaptive machine learning model generation, data storage, etc. The frontend component sends requests to the backend, highlights target units, presents candidate suggestions, sends user interaction to the database and so on. The API component transforms the frontend requests to the backend and returns responses to the frontend. Figure 1 shows the three main components of PAR4SEM and their interactions.

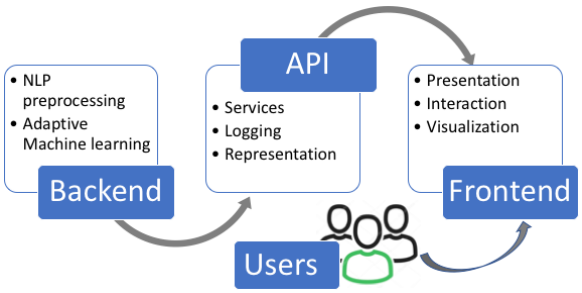

Figure 1: The main components of PAR4SEM

2.1 The Backend Component

The backend component consists of several modules. For the adaptive paraphrasing system, the first component is to identify possible target units (such as single words, phrases, or multi-word expressions). For our lexical simplification use-case, the target units identification component is instantiated with the datasets obtained from Yimam et al. (2017a,b, 2018). The adaptive target identification unit then keeps on learning from the usage data (when the user highlights portions of the text to get paraphrase candidate suggestions).

Once target units are marked or recognized (by the target unit identification system), the next step is to generate possible candidate suggestion for the target unit (paraphrase candidates). The candidate suggestion module includes candidate generation and candidate ranking sub-modules. Section 2.1.1 discusses our approaches to generating and ranking paraphrase candidates in detail.

2.1.1 Paraphrasing Resources

Paraphrase resources are datasets where target units are associated with a list of candidate units equivalent in meaning, possibly ranked by their meaning similarity. One can refer to the work of Ho et al. (2014) about the details on how paraphrase resources are produced, but we will briefly discuss the different types of paraphrase resources that are used in generating candidate suggestions for PAR4SEM.

PPDB 2.0: The Paraphrase Database (PPDB) is a collection of over 100 million paraphrases that was automatically constructed using a bilingual pivoting method. Recently released PPDB 2.0 includes improved paraphrase rankings, en-

[1] `https://uhh-lt.github.io/par4sem/`

tailment relations, style information, and distributional similarity measures for each paraphrase rule (Pavlick et al., 2015).

WordNet: We use WordNet synonyms, which are described as *words that denote the same concept and are interchangeable in many contexts* (Miller, 1995), to produce candidate suggestions for a given target unit.

Distributional Thesaurus – JoBimText: We use JoBimText, an open source platform for large-scale distributional semantics based on graph representations (Biemann and Riedl, 2013), to extract candidate suggestions that are semantically similar to the target unit.

Phrase2Vec: We train a Phrase2Vec model (Mikolov et al., 2013) using English Wikipedia and the AQUAINT corpus of English news text (Graff, 2002). Mikolov et al. (2013) pointed out that it is possible to extend the word based embeddings model to phrase-based model using a data-driven approach where each phrase or multi-word expressions are considered as individual tokens during the training process. We have used a total of 79,349 multiword expression and phrase resources as given in Yimam et al. (2016). We train the Phrase2Vec embeddings with 200 dimensions using skip-gram training and a window size of 5. We have retrieved the top 10 similar words to the target units as candidate suggestions.

2.1.2 Adaptive Machine Learning

PAR4SEM incorporates two adaptive machine learning models. The first one is used to identify target units (*target adaption*) in a text while the second one is used to rank candidate suggestions (*ranking adaption*). Both models make use of usage data as a training example. The target adaption model predicts target units based on the usage data (training examples) and sends them to the frontend component, which are then highlighted for the user. If the user replaced the highlighted target units, they are considered as positive training examples for the next iteration.

The ranking adaption model first generates candidate paraphrases using the paraphrase resource datasets (see Section 2.1.1). As all the candidates generated from the paraphrase resources might not be relevant to the target unit at a context, or as the number of candidates to be displayed might be excessively large (for example the PPDB 2.0 resource alone might produce hundreds of candidates for a target unit), we re-rank the candidate suggestions using a learning-to-rank adaptive machine learning model. Figure 2 displays the process of the adaptive models while Figure 3 displays the pipeline (as a loop) used in the generations of the adaptive models.

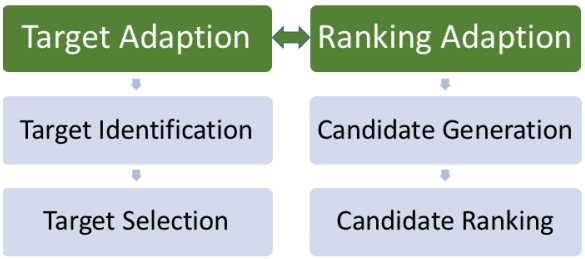

Figure 2: The main and sub-processes of target and ranking adaption components of PAR4SEM.

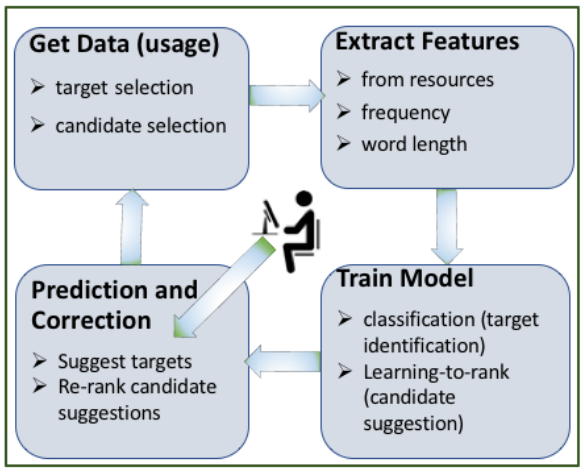

Figure 3: The loop for the generation of the adaptive models of PAR4SEM.

The whole process is iterative, interactive, and adaptive in a sense that the underlying models (both target adaption and ranking adaption) get usage data continuously from the user. The models get updated for each iteration, where n examples conducted in a batch mode without model update, and provide better suggestions (as target units or candidate suggestions) for the next iteration. The user interacts with the tool, probably accepting or rejecting tool suggestions, which is fed as a training signal for the next iterations model. Figure 4 shows the entirety of interactions, iterations, and adaptive processes of the PAR4SEM system. In the first iteration, the ranking is provided using a baseline language model while for the subsequent iterations, the usage data from the previous batches (t-1) is used to train a model that is used to rank the current batch (t).

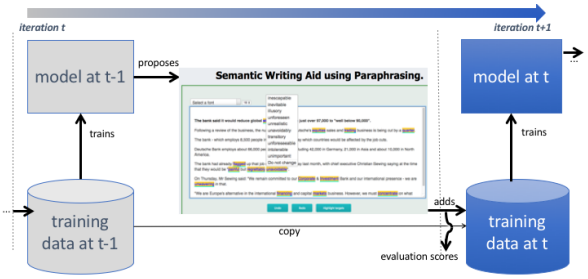

Figure 4: The iterative and adaptive interaction of PAR4SEM.

2.1.3 Backend Technologies

The backend components are fully implemented using the Java programming language. Text segmentation such as sentence splitting, tokenization, lemmatization, and parts of speech tagging is handled using the Apache OpenNLP[2] library.

For the target unit identification system, we have used *Datumbox*[3], a powerful open-source machine learning framework written in Java. We have used specifically the *Adaboost* classification algorithm.

For the ranking model, RankLib, which is the well-known library for the learning to rank algorithms from the Lemur[4] project is integrated. All the data related to PAR4SEM interactions (usage data, time, and user details) are stored in a MySQL database.

2.2 Frontend Components

The frontend component of PAR4SEM is designed where document composing with a semantic paraphrasing capability is integrated seamlessly. It is a web-based application allowing access either on a local installation or over the internet.

2.2.1 UI Components for Paraphrasing

The frontend part of PAR4SEM comprises different modules. The most important UI component is the text editing interface (Figure 5) that allows for adding text, highlighting target units, and displaying candidate suggestions. (1) is the main area to compose (or paste) texts. The operational buttons ((2)) are used to perform some actions such as to undo and redo (composing, target unit highlighting, and paraphrase ranking), automatically highlighting target units, and clear the text area. Target units are underlined in cyan color and highlighted in yellow background color as a link ((3)) which enables users to click, display, and select candidate suggestions for a replacement ((4)).

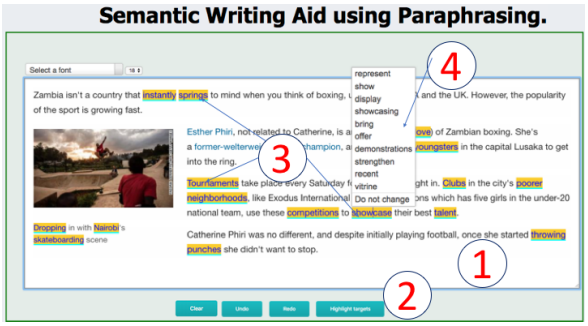

Figure 5: The PAR4SEM text editing component that is used to compose texts, highlight target units, and display candidate suggestions for the target units.

2.2.2 Frontend Technologies

The frontend components are implemented using HTML, CSS and JavaScript technologies. For the text highlighting and candidate suggestion replacement, the jQuery Spellchecker[5] module is slightly modified to incorporate the semantic highlighting (underline in cyan and a yellow background). The accompanied documentation and datasets of PAR4SEM[6] are hosted at Github pages.

2.3 RESTful API Component

Semantic technologies, those like PAR4SEM incorporates highly dynamic dimensions. One dimension is that the paraphrase resources can be varied depending on the need of the application. Another dimension is that the application can be in different languages. If the backend and the frontend technologies are highly coupled, it will be difficult to reuse the application for different languages, resources, and applications. To solve this problem, we have developed PAR4SEM using a RESTful API (aka. microservices) as a middleware between the backend and the frontend components.

The API component consumes requests (getting target units and candidate suggestions) or resources (saving usage data such as selection of new target units, user's preference for candidate ranking, user and machine information) from the frontend and transfers them to the backend. The

[2] https://opennlp.apache.org/
[3] http://www.datumbox.com/
[4] https://sourceforge.net/p/lemur/wiki/RankLib/
[5] http://jquery-spellchecker.badsyntax.co/
[6] https://uhh-lt.github.io/par4sem/

backend component translates the requests or resources and handles them accordingly. Spring Boot[7] is used to implement the API services.

2.3.1 Installation and Deployment

As PAR4SEM consists of different technologies, machine learning setups, resources, and configurations, we opted to provide a Docker-based installation and deployment options. While it is possible to fully install the tool on ones own server, we also provide an API access for the whole backend services. This allows users to quickly and easily install the frontend component and relay on our API service calls for the rest of the communications.

3 Use-case – Adaptive Text Simplification using Crowdsourcing

An appropriate use case for adaptive paraphrasing is lexical text simplification. Lexical simplification aims to reduce the complexity of texts due to difficult words or phrases in the text (Siddharthan, Advaith, 2014). We have used PAR4SEM particularly for text simplification task with an emphasis of making texts accessible for language learners, children, and people with disabilities.

We conducted the experiment by integrating the tool into the Amazon Mechanical Turk (MTurk)[8] crowdsourcing and employ workers to simplify texts using the integrated adaptive paraphrasing system. While PAR4SEM is installed and run on our local server, we make use of the MTurk's external HIT functionality to embed and conduct the text simplification experiment. Once workers have access to our embedded tool in the MTurk browser, they will be redirected to our local installation to complete the simplification task. Figure 5 shows the PAR4SEM user interface to perform text simplification task by the workers while Figure 7 shows the instructions as they appeared inside the MTurk's browser.

We asked workers to simplify the text for the target readers, by using the embedded paraphrasing system. Difficult words or phrases are automatically highlighted so that workers can click and see possible candidate suggestions. The experiment was conducted over 9 iterations, where the ranking model is updated using the training dataset (usage data) collected in the previous iterations. The first iteration does not use ranking model but

[7] https://projects.spring.io/spring-boot/

[8] https://www.mturk.com/

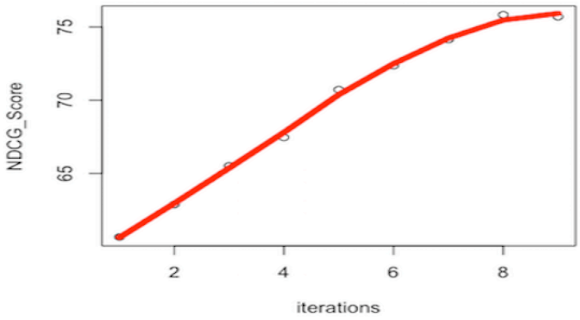

Figure 6: Learning curve showing the increase of NDCG@10 score over 9 iterations.

candidates are presented using a default language-model-based ranking. In (Yimam and Biemann, 2018) we have shown that the adaptive paraphrasing system adopts very well to text simplification, improving the NDCG (Wang et al., 2013) score from 60.66 to 75.70. Figure 6 shows the learning curve for the different iterations conducted in the experiment.

4 Conclusion

In this paper, we have described PAR4SEM, a semantic writing aid tool based on an embedded adaptive paraphrasing system. Unlike most annotation tools, which are developed exclusively to collect training examples for machine learning applications, PAR4SEM implements an adaptive paraphrasing system where training examples are obtained from usage data.

To the best of our knowledge, PAR4SEM is the first of its kind where machine learning models are improved based on usage data and user feedback (correction of suggestions) for semantic applications. PAR4SEM is used in a text simplification use-case. Evaluation of the system showed that the adaptive paraphrasing system for text simplification successfully adapted to the target task in a small number of iterations.

For future work, we would like to evaluate the system in an open task setting where users can paraphrase resp. simplify self-provided texts, and explore how groups of similar users can be utilized to provide adaptations for their respective sub-goals.

References

Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! a framework for lexical expansion with contextual similarity. *JLM*, 1(1):55–95.

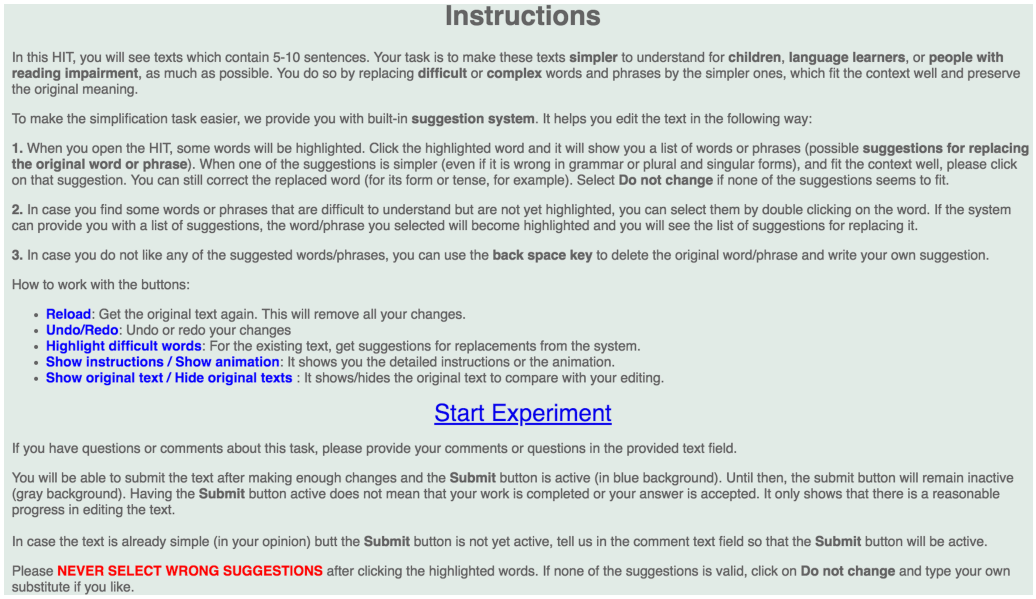

Figure 7: The instructions for the text simplification task using PAR4SEM

Michael Denkowski, Alon Lavie, Isabel Lacruz, and Chris Dyer. 2014. Real Time Adaptive Machine Translation for Post-Editing with cdec and TransCenter . In *Proc. EACL 2014 Workshop on HCAT*, pages 72–77, Gothenburg, Sweden.

David Graff. 2002. The AQUAINT Corpus of English News Text LDC2002T31. In *Web Download. Philadelphia: Linguistic Data Consortium.*

ChukFong Ho, Masrah Azrifah Azmi Murad, Shyamala Doraisamy, and Rabiah Abdul Kadir. 2014. Extracting lexical and phrasal paraphrases: a review of the literature. *Artificial Intelligence Review*, 42(4):851–894.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *ANIPS/ACNIPS*, pages 3111–3119, Stateline, Nevada, USA.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM*, 38(11):39–41.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proc. ACL/IJCNLP*, pages 425–430, Beijing, China.

Jyh-Jian Sheu, Ko-Tsung Chu, Nien-Feng Li, and Cheng-Chi Lee. 2017. An efficient incremental learning mechanism for tracking concept drift in spam filtering. *PLOS ONE*, 12(2):1–17.

Siddharthan, Advaith. 2014. A survey of research on text simplification. *IJAL*, 165(2):259–298.

Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, and Tie-Yan Liu. 2013. A Theoretical Analysis of Normalized Discounted Cumulative Gain (NDCG) Ranking Measures. In *MLR*, pages 25–54, Princeton, New Jersey, USA.

Seid Muhie Yimam and Chris Biemann. 2018. Par4sim – adaptive paraphrasing for text simplification. In *Proc. of COLING 2018*, pages 331–342, Santa Fe, New Mexico, USA.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo Paetzold, Lucia Specia, Sanja Štajner, Anaïs Tack, and Marcos Zampieri. 2018. A report on the complex word identification shared task 2018. In *Proc. of the 13th BEA*, pages 66–78, New Orleans, Louisiana.

Seid Muhie Yimam, Héctor Martínez Alonso, Martin Riedl, and Chris Biemann. 2016. Learning Paraphrasing for Multiword Expressions. In *Proc. of the 12th Workshop on MWE*, pages 1–10, Berlin, Germany.

Seid Muhie Yimam, Sanja Štajner, Martin Riedl, and Chris Biemann. 2017a. CWIG3G2 - Complex Word Identification Task across Three Text Genres and Two User Groups. In *Proc. IJCNLP-17*, pages 401–407, Taipei, Taiwan.

Seid Muhie Yimam, Sanja Štajner, Martin Riedl, and Chris Biemann. 2017b. Multilingual and cross-lingual complex word identification. In *PROC. of RANLP*, pages 813–822, Varna, Bulgaria. INCOMA Ltd.

Dani Yogatama, Chong Wang, Bryan R. Routledge, Noah A. Smith, and Eric P. Xing. 2014. Dynamic language models for streaming text. *TACL*, 2:181–192.