CDM: Combining Extraction and Generation for Definition Modeling

Jie Huang  Hanyin Shao  Kevin Chen-Chuan Chang
University of Illinois at Urbana-Champaign, USA
{jeffhj, hanyins2, kcchang}@illinois.edu

Abstract

Definitions are essential for term understanding. Recently, there is an increasing interest in extracting and generating definitions of terms automatically. However, existing approaches for this task are either extractive or abstractive—definitions are either extracted from a corpus or generated by a language generation model. In this paper, we propose to combine extraction and generation for definition modeling: first extract self- and correlative definitional information of target terms from the Web and then generate the final definitions by incorporating the extracted definitional information. Experiments demonstrate our framework can generate high-quality definitions for technical terms and outperform state-of-the-art models for definition modeling significantly.\(^1\)

1 Introduction

Definitions of terms are highly summarized sentences that capture the main characteristics of terms. To understand a term, the most straightforward way is to read its definition. Recently, acquiring definitions of terms automatically has aroused increasing interest. There are two main approaches: extractive, corresponding to definition extraction, where definitions of terms are extracted from existing corpora automatically (Anke and Schockaert, 2018; Veyseh et al., 2020; Kang et al., 2020); and abstractive, corresponding to definition generation, where definitions of terms are generated conditioned with the target terms and their contexts (Noraset et al., 2017; Gadetsky et al., 2018; Bevilaqua et al., 2020).

However, both extractive and abstractive approaches have their limitations. For instance, extracting high-quality definitions for terms would be difficult due to the incompleteness and low quality of data sources. Generating definitions for terms would be challenging if terms are technical (e.g., need domain knowledge to understand) while the contexts cannot provide sufficient knowledge. Consequently, existing models perform poorly on technical terms. In our human evaluation, we find most definitions produced by the state-of-the-art abstractive model contain wrong information.

Fortunately, definition extraction and definition generation can complement each other. On one hand, definition generator has the potentials to help the extractor by refining and synthesizing the extracted definitions; on the other hand, definition extractor can retrieve useful definitional information as knowledge for the generator to produce definitions. However, surprisingly, existing works are either extractive or abstractive, even do not connect and compare them.

Therefore, in this work, we propose to combine definition extraction and definition generation for definition modeling. We achieve this by introducing a framework consisting of two processes: extraction, where definitional information of terms are extracted from the Web; and generation, where the final definitions are generated with the help of the extracted definitional information.

We build models for extraction and generation based on Pre-Trained Language Models (Devlin et al., 2019; Lewis et al., 2020; Brown et al., 2020). Specifically, for extraction, we propose a BERT-based definition extractor to extract self-definitional information (i.e., definitional sentences of the target term). We also suggest that related terms can help defining the target term and leverage Wikipedia as the external knowledge source to retrieve correlative definitional information (i.e., definitions of related terms). For generation, we design a BART-based definition generator to produce the final definition by incorporating the extracted knowledge.

From another perspective, we propose to reform the problem of definition modeling, which is previously mainly defined as generating definitions of terms conditioned with a target term and a given

\(^1\)Code and data are available at https://github.com/jeffhj/CDM.
Instead, we restudy this problem as defining terms with extracted knowledge. This setting is in line with human behavior: to understand a term, compared to reading the given sentence it is used in, it is more straightforward and helpful to search and read its relevant content on the Internet.

Our framework for definition modeling is simple and flexible that can easily be further expanded by leveraging more advanced language models. Experimental results demonstrate our simple model outperforms state-of-the-art models significantly (e.g., BLEU score from 8.76 to 22.66, human annotated score from 2.34 to 4.04), with several interesting findings: 1) for computer science terms, our extractive model can achieve performance comparable to (even better than) state-of-the-art abstractive models; 2) both self- and correlative definitional information are significant to define a term; 3) the quality of definitions generated by our best model is high, while the state-of-the-art models suffer severely from hallucinations, i.e., generating irrelevant or contradicted facts.

Our contributions are summarized as follows:
• As far as we know, we are the first to connect and combine definition extraction and definition generation—a simple idea that can significantly improve the performance of definition modeling.
• We propose to restudy definition modeling as generating definitions of terms with extracted knowledge.
• We design a novel framework for definition modeling by incorporating both self- and correlative definitional information of terms.
• We publish two datasets for technical terms, along with definitions of ~75,600 computer science terms generated by our model.

2 Related Work

Definition Extraction. Existing works for definition extraction can be roughly divided into three categories: 1) rule-based, which extracts definitions with defined linguistic rules and templates (Klavans and Muresan, 2001; Cui et al., 2004; Fahmi and Bouma, 2006); 2) machine learning-based, which extracts definitions by statistical machine learning with carefully designed features (Westerhout, 2009; Jin et al., 2013); 3) deep learning-based, the state-of-the-art approach for definition extraction, which is based on deep learning models such as CNN, LSTM, and BERT (Anke and Schockaert, 2018; Veyseh et al., 2020; Kang et al., 2020).

Definition Generation. Definition generation, or definition modeling, was first introduced in (Noraset et al., 2017), which aims to generate definitions of words with word embeddings. Later works on definition generation put more emphasis on generating definitions of words/phrases with given contexts (Gadetsky et al., 2018; Ishiwatari et al., 2019; Washio et al., 2019; Li et al., 2020; Reid et al., 2020; Bevilacqua et al., 2020). There are also recent works on definition modeling for other languages, e.g., Chinese, by incorporating the special properties of the specific language (Yang et al., 2020; Zheng et al., 2021).

3 Methodology

Our framework for definition modeling consists of two processes: extraction, which extracts self- and correlative definitional information of target terms from the Web; and generation, which generates the final definitions by incorporating the extracted definitional information.

3.1 Extraction

3.1.1 Self-Definitional Information

To define a term, it is natural to refer to definitional sentences containing the target term, named Self-Definitional Information (SDI). We achieve SDI by first extracting sentences containing the target term from the Web (more details are in Section 4.1) and then using a classifier to rank the extracted sentences.

To build the classifier, we apply the BERT model (Devlin et al., 2019), which has achieved excellent results on various text classification tasks. We adopt a simple encoding scheme, which is “[CLS] term [DEF] sentence”, e.g., “[CLS] machine learning [DEF] machine learning is the study of computer algorithms that improve automatically through experience and by the use of data.” The final hidden state of the first token [CLS] is used as the representation of the whole sequence and a classification layer is added. By fine-tuning on the term-sentence pairs, the model can learn to distinguish whether the sentence is the definition of the target term. SDI is then obtained as the top definitions by ranking the sentences according to the confidence of the prediction. We refer to this model as SDI-Extractor.

3.1.2 Correlative Definitional Information

To understand a term, in addition to utilizing SDI, we can also refer to the definitions of its related
terms, i.e., Correlative Definitional Information (CDI). For instance, to define few-shot learning, we can incorporate definitions of zero-shot learning and meta learning, with which we can know the meaning of “shot” and “learning” and may define few-shot learning similarly to zero-shot learning.

To get related terms and their definitions, we leverage Wikipedia as the external knowledge source, which covers a wide range of domains and contains high-quality definitions for a large number of terms. Specifically, we follow the core-fringe notion in (Huang et al., 2021), where core terms are terms that have corresponding Wikipedia pages, and fringe terms are ones that are not associated with a Wikipedia page. For each term (both core and fringe), we treat it as query to retrieve the most relevant core terms via document ranking based on Elasticsearch (Gormley and Tong, 2015), and extract first sentences on the corresponding Wikipedia pages as the definitions of related terms. We refer to this model as CDI-Extractor.

3.2 Generation

After extraction, we get the self- and correlative definitional information of terms. This kind of information captures important characteristics of terms and can be further refined and synthesized into the final definition by a definition generator.

Definition generation can be formulated as a conditioned sentence generation task—generating a coherent sentence to define the target term. Formally, we apply the standard sequence-to-sequence formulation: given term \( x \), combining with the extracted sentences \( S_s \) (for SDI) and \( S_c \) (for CDI), the probability of the generated definition \( d \) is computed auto-regressively:

\[
P(d|x, S_s, S_c) = \prod_{i=1}^{m} P(d_i|d_{0:i-1}, x, S_s, S_c),
\]

where \( m \) is the length of \( d \), \( d_i \) is the \( i \)th token of \( d \), and \( d_0 \) is a special start token.

Following (Bevilacqua et al., 2020), to build the generator, we employ a recently proposed transformer-based encoder-decoder model, i.e., BART (Lewis et al., 2020), which is pre-trained and can be fine-tuned to perform specific conditional language generation tasks with specific training input-output pairs. Different from existing works (Gadetsky et al., 2018; Ishiwatari et al., 2019; Bevilacqua et al., 2020) which aim to learn to define a term in a given context, we propose to learn to define a term using the extracted knowledge. Specifically, we aim to fine-tune the BART model to generate the definition of the target term based on the distributed representation of the term and the extracted definitional information.

To apply the BART model, for a target term, we adopt the following encoding scheme: “term [DEF] sent_1 [SEP] sent_2 ... [SEP] sent_k [DEF] sent_1’ [SEP] sent_2’ ... [SEP] sent_k’”, where \( sent_i \) and \( sent_i’ \) are the \( i \)th sentences ranked by SDI-Extractor and CDI-Extractor, respectively. We fine-tune BART to produce the ground-truth definition conditioned with the encoded input.

After training, given a new term, we get corresponding SDI and CDI according to Section 3.1. We encode the term and the top \( k \) ranked sentences of SDI and top \( k’ \) ranked sentences of CDI as described above and use the generator to produce the final definition. We refer to this model as CDM-Sk,Ck’, i.e., Combined Definition Modeling.

4 Experiments

4.1 Datasets

Existing datasets for definition modeling are mainly for general words/phrases. In this paper, we focus our evaluation on technical terms—which are comparatively less studied (neglected) in existing works. Compared to general words/phrases, technical terms are less ambiguous but more professional, i.e., a technical term usually only has one meaning, but it requires domain knowledge to understand.

Definition Extraction. We build a dataset for definition extraction by Wikipedia. We first collect terms with Wikipedia Category. Specifically, we traverse from three root categories, including Category:Subfields of computer science\(^2\), Category:Fields of mathematics\(^3\), and Category:Subfields of physics\(^4\), and collect pages at the first three levels of the hierarchies. For each page, we process the title with lemmatization as the term, extract the first sentence in the summary section as the corresponding definition, and sample \( \leq 5 \) sentences containing the target term from other sections as negatives. We filter out terms with

\(^2\)https://en.wikipedia.org/wiki/Category:Subfields_of_computer_science
\(^3\)https://en.wikipedia.org/wiki/Category:Fields_of_mathematics
\(^4\)https://en.wikipedia.org/wiki/Category:Subfields_of_physics
frequency < 5 in the arXiv corpus\(^5\). The dataset contains 26,559 positive and 121,975 negative examples, and the train/dev/test split is 0.8/0.1/0.1.

**Definition Generation.** We focus on generating definitions for computer science terms since the audience of this paper will be more interested in and familiar with them. We collect term candidates (author-assigned keywords) by web scraping from Springer publications on computer science. We filter out terms with frequency < 5. For each term in the list, URLs of the top 20 results from Google search are visited. Then the sentences containing the target term are extracted. After crawling, there are 75,690 terms having candidate sentences. Among these terms, 16,759 have a corresponding Wikipedia page. We extract the first sentence on each page as the ground-truth definition. We use these terms to build a dataset for generation. The train/dev/test split is 0.7/0.1/0.2, where terms in the dev and test sets do not overlap with those in the train set for definition extraction.

### 4.2 Experimental Setup

**Baselines.** For extraction, we compare SDI-Extractor with a CNN baseline and a CNN-BiLSTM baseline proposed in (Anke and Schockaert, 2018). Here we should mention that the more recent models (Veyseh et al., 2020; Kang et al., 2020) cannot be compared directly since these works focus on a fine-grained sequence labeling task, where the training data also requires additional labeling. Besides, extraction is not the focus of this paper; therefore, we put more emphasis on the evaluation for generation.

For generation, we evaluate on two most recent models on definition modeling for English phrases/terms: Generationary (also based on BART) (Bevilacqua et al., 2020) and VCDM (Reid et al., 2020), an extractive baseline, and several variants of our proposed model:

- **Gen (w/o context):** A simple version of Generationary, which generates definitions of terms without any context provided.
- **Gen (w/ context):** Generationary with a sentence containing the target term as context.
- **VCDM:** The VCDM baseline, where context is provided by a sentence containing the target term.
- **Extractive:** An extractive baseline, which outputs the candidate definition with the highest confidence score predicted by SDI-Extractor (Section 3.1.1).

- **CDM-S\(k\), C\(k\):** The combined definition modeling model introduced in Section 3.2. \(S_k\) or \(C_k\) is omitted when \(k\) or \(k\) is equal to 0.

For all the input, we exclude sentences from Wikipedia to avoid the models from seeing the ground truth.

**Metrics.** For extraction, we use the standard precision, recall, and F1 scores to evaluate the performance. For generation, we follow (Bevilacqua et al., 2020) and apply several automatic metrics, including BLEU (BL) (Papineni et al., 2002), ROUGE-L (R-L) (Lin, 2004), METEOR (MT) (Banerjee and Lavie, 2005), and BERTScore (BS) (Zhang et al., 2019). We also ask three human annotators to evaluate the output definitions for 50 terms randomly selected from the test set, with a 1-5 rating scale used in (Ishiwatari et al., 2019): 1) completely wrong or self-definition; 2) correct topic with wrong information; 3) correct but incomplete; 4) small details missing; 5) correct.

**Implementation Details.** For extraction, we adopt BERT-base-uncased from huggingface transformers framework (Wolf et al., 2020). We use the default hyperparameters and fine-tune the model using Adam (Kingma and Ba, 2015) with learning rate of \(2 \times 10^{-6}\). For generation, we employ the fairseq library\(^6\) to build the BART generator and adopt the hyperparameters and settings as suggested in (Bevilacqua et al., 2020). All the models were trained on a single NVIDIA Quadro RTX 5000 GPU. The training of CDM can be finished in one hour.

### 4.3 Definition Extraction

Table 1 reports the results of definition extraction. We observe SDI-Extractor outperforms baselines significantly and the performance is quite satisfactory, which means our definition extractor can extract useful self-definitional information for terms.

|                | Precision | Recall | F1   |
|----------------|-----------|--------|------|
| CNN            | 91.84     | 90.66  | 91.25|
| C-BLSTM        | 91.59     | 88.93  | 90.24|
| SDI-Extractor  | 96.72     | 97.67  | 97.19|

Table 1: Results of definition extraction.

\(^5\)https://www.kaggle.com/Cornell-University/arxiv

\(^6\)https://github.com/pytorch/fairseq/tree/master/examples/bart
|          | BL | R-L | MT | BS |
|----------|----|-----|----|----|
| Extractive | 15.62 | 29.41 | 16.41 | 79.02 |
| Gen (w/o context) | 8.31 | 28.02 | 12.83 | 77.97 |
| Gen (w/ context) | 8.76 | 30.00 | 13.15 | 78.73 |
| VCDM | 8.27 | 29.24 | 12.77 | 78.23 |
| CDM-C5 | 12.26 | 29.90 | 14.55 | 79.09 |
| CDM-S1 | 17.12 | 34.67 | 17.46 | 80.75 |
| CDM-S3 | 19.08 | 35.48 | 18.44 | 81.16 |
| CDM-S5 | 20.21 | 35.98 | 19.06 | 81.33 |
| CDM-S10 | 19.27 | 36.34 | 18.79 | 81.51 |
| CDM-S5,C5 | 22.66 | 38.12 | 20.30 | 82.00 |

Table 2: Results of definition generation on automatic metrics. The best models are bold and second best ones are underlined.

|          | Score (1-5) |
|----------|-------------|
| Extractive | 3.57 |
| Gen (w/ context) | 2.34 |
| CDM-S1 | 3.65 |
| CDM-S5 | 3.99 |
| CDM-S5,C5 | 4.04 |

Table 3: Averaged human annotated scores.

4.4 Definition Generation

**Automatic Evaluation.** Table 2 shows the results on automatic metrics. We observe the proposed CDM model outperforms Generationary and VCDM significantly. Comparing Gen (w/ context) with Gen (w/o context), we find contexts (random sentences containing the target terms) only help slightly for definition modeling for technical terms. Besides, CDM-S5 outperforms CDM-S3, while CDM-S3 outperforms CDM-S1, which means the sentences extracted by Extractor-SDI can provide important definitional information. Among all the models, CDM-S5,C5 achieves the best performance, which demonstrates SDI and CDI are both significant to define terms.

An interesting finding is that our simple extractive model also outperforms the baselines. We suppose this is because technical terms are difficult to define without external knowledge. For instance, it is almost impossible for a model to generate the definition for “twin prime” only with context “proof of this conjecture would also imply the existence an infinite number of twin primes”, while the definition can possibly be retrieved from the Web.

**Human Evaluation.** Table 3 reports the human evaluation results. We observe the state-of-the-art baseline Gen (w/ context) is difficult to generate reasonable definitions for technical terms. In contrast, the proposed CDM-S5,C5 model can produce high-quality definitions in most cases.

4.5 Sensitivity to Term Frequency

To investigate the sensitivity of the models with respect to the popularity of terms, we report the results according to term frequency in Figure 1. We observe Generationary achieves slightly worse performance for less popular terms on all metrics, while CDM performs well for low-frequency terms, which indicates our framework can produce high-quality definitions for long-tail terms. We suppose this is because, although long-tail terms are less frequent, we can still extract useful definitional information from the entire Web and incorporate it to define terms.

4.6 Generation Examples

In Table 4, we show some sample outputs of three models: Extractive, Gen (w/ context), and CDM-S5,C5, with ground-truth definitions in Wikipedia (Gold) as references.

From the results, we observe although the extractive baseline can produce reasonable sentences, the output sentences may not be high-quality definitional sentences of the target terms. We also find Gen (w/ context) suffers severely from hallucinations, i.e., generating irrelevant or contradicted facts. This is mainly because the contexts of terms may not provide sufficient knowledge to define technical terms. In contrast, the quality of definitions generated by CDM-S5,C5 is high— all the generated definitions capture the main characteristics of the target terms correctly.
Table 4: Sample of definitions produced by Extractive, Gen (w/ context), and CDM-S5,C5.

|                      | Extractive | Gen (w/ context) | CDM-S5,C5 | Gold                      |
|----------------------|------------|------------------|-----------|---------------------------|
| intelligent user interfaces | ACM IUI 2021 is the 26th annual premier international forum for reporting outstanding research and development on intelligent user interfaces. | An intelligent user interface (UI) is a user interface that has been developed by a group of researchers interested in enabling natural human-computer interaction by combining techniques from computer vision, machine learning, computer graphics, human-computer interaction and psychology. | An intelligent user interface (Intelligent UI,IUI, or sometimes Interface Agent) is a user interface (UI) that involves some aspect of artificial intelligence (AI or computational intelligence). |
| wear leveling | The term preemptive wear leveling (PWL) has been used by Western Digital to describe their preservation technique used on hard disk drives (HDDs) designed for storing audio and video data. | Wear leveling is the process of reducing the wear of a Flash die. | Wear leveling is a technique used to increase the lifetime of a solid-state drive (SSD). | Wear leveling (also written as wear leveling) is a technique for prolonging the service life of some kinds of erasable computer storage media, such as flash memory, which is used in solid-state drives (SSDs) and USB flash drives, and phase-change memory. |
| gittins index | In applied mathematics, the Gittins index is a real scalar value associated to the state of a stochastic process with a reward function and with a probability of termination. | The Gittins index is a decision-making tool used in decision-making and project management. | In applied mathematics, the Gittins index is a real scalar value associated to the state of a stochastic process with a reward function and with a probability of termination. | The Gittins index is a measure of the reward that can be achieved through a given stochastic process with certain properties, namely: the process has an ultimate termination state and evolves with an option, at each intermediate state, of terminating. |
| reduplication | The term compensatory reduplication refers to duplication that serves a phonological purpose. | In mathematics, reduplication is a generalization of the concept of reduplication. | Replication is the repetition of an entire word, word stem (root with one or more affixes), or root. | In linguistics, reduplication is a morphological process in which the root or stem of a word (or part of it) or even the whole word is repeated exactly or with a slight change. |
| power delay profile | The power delay profile of a channel represents the average power of the received signal in terms of the delay with respect to the first arrival path in multi-path transmission. | A power delay profile (PDP) is a measure of the time delay between the transmission and reception of a signal. | In telecommunications, the power delay profile (PDP) gives the intensity of a signal received through a multipath channel as a function of time delay. | The power delay profile (PDP) gives the intensity of a signal received through a multipath channel as a function of time delay. |

5 Conclusion

In this paper, we combine extraction and generation for definition modeling. We show that, by incorporating extracted self- and correlative definitional information, the generator can produce high-quality definitions for technical terms. Experimental results demonstrate the effectiveness of our framework. As future work, we plan to apply our methods to more domains and construct several online domain dictionaries.

References

Luis Espinosa Anke and Steven Schockaert. 2018. Syntactically aware neural architectures for definition extraction. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 378–385.

Santanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation metrics for machine translation and/or summarization, pages 65–72.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generational or:“how we went beyond word sense inventories and learned to gloss”. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7207–7221.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Saxty, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCallum, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc.

Hang Cui, Min-Yen Kan, and Tat-Seng Chua. 2004. Unsupervised learning of soft patterns for generating definitions from online news. In Proceedings of the 13th international conference on World Wide Web, pages 90–99.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.
Ismail Fahmi and Gosse Bouma. 2006. Learning to identify definitions using syntactic features. In Proceedings of the Workshop on Learning Structured Information in Natural Language Applications.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 266–271.

Clinton Gormley and Zachary Tong. 2015. Elastic-search: the definitive guide: a distributed real-time search and analytics engine. " O'Reilly Media, Inc.”.

Jie Huang, Kevin Chen-Chuan Chang, Jinjun Xiong, and Wen-mei Hwu. 2021. Measuring fine-grained domain relevance of terms: A hierarchical corefringe approach. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing.

Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. 2019. Learning to describe unknown phrases with local and global contexts. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3467–3476.

Yiping Jin, Min-Yen Kan, Jun Ping Ng, and Xiangnan He. 2013. Mining scientific terms and their definitions: A study of the acl anthology. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 780–790.

Dongyeop Kang, Andrew Head, Risham Sidhu, Kyle Lo, Daniel S Weld, and Marti A Hearst. 2020. Document-level definition detection in scholarly documents: Existing models, error analyses, and future directions. In Proceedings of the First Workshop on Scholarly Document Processing, pages 196–206.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. Proceedings of the 3rd International Conference on Learning Representations.

Judith L Klavans and Smaranda Muresan. 2001. Evaluation of the defnder system for fully automatic glossary construction. In Proceedings of the AMIA Symposium, page 324. American Medical Informatics Association.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880.

Jiahuan Li, Yu Bao, Shujian Huang, Xinyu Dai, and CHEN Jiajun. 2020. Explicit semantic decomposition for definition generation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 708–717.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Text summarization branches out, pages 74–81.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 31.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages 311–318.

Machel Reid, Edison Marrese-Taylor, and Yutaka Matsuo. 2020. Vcdm: Leveraging variational bi-encoding and deep contextualized word representations for improved definition modeling. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6331–6344.

Amir Veyseh, Franck Dernoncourt, Dejing Dou, and Thien Nguyen. 2020. A joint model for definition extraction with syntactic connection and semantic consistency. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 9098–9105.

Koki Washio, Satoshi Sekine, and Tsuneaki Kato. 2019. Bridging the defined and the defining: Exploiting implicit lexical semantic relations in definition modeling. 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 3521–3527.

Eline Westerhout. 2009. Definition extraction using linguistic and structural features. In Proceedings of the 1st Workshop on Definition Extraction, pages 61–67.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 38–45.

Liner Yang, Cunliang Kong, Yun Chen, Yang Liu, Qinan Fan, and Erhong Yang. 2020. Incorporating sememes into chinese definition modeling.
Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In International Conference on Learning Representations.

Hua Zheng, Damai Dai, Lei Li, Tianyu Liu, Zhifang Sui, Baobao Chang, and Yang Liu. 2021. Decompose, fuse and generate: A formation-informed method for chinese definition generation. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5524–5531.