A Unified Model for Reverse Dictionary and Definition Modelling

Pinzhen Chen **Zheng Zhao**
School of Informatics, University of Edinburgh
{pinzhen.chen, zheng.zhao}@ed.ac.uk

Abstract

We train a dual-way neural dictionary to guess words from definitions (reverse dictionary), and produce definitions given words (definition modelling). Our method learns the two tasks simultaneously, and handles unknown words via embeddings. It casts a word or a definition to the same representation space through a shared layer, then generates the other form from there, in a multi-task fashion. The model achieves promising automatic scores without extra resources. Human annotators prefer the proposed model's outputs in both reference-less and reference-based evaluation, which indicates its practicality. Analysis suggests that multiple objectives benefit learning.

1 Introduction

A monolingual dictionary is a collection of words paired with their definitions on a large scale. The main use of such a resource is to find a word or a definition having known the other. Formally, the task of generating a textual definition given a word is called *definition modelling*; on the contrary, the task of retrieving a word given a definition is called *reverse dictionary*. Lately, the two tasks are approached using neural networks (Hill et al., 2016; Noraset et al., 2017); in turn, the tasks help researchers better understand word sense and embeddings. This can also benefit low-resource languages where high-quality dictionaries are not available (Yan et al., 2020). Moreover, there are practical applications including language education, paraphrasing, semantic search, etc.

While previous work solves one problem at a time, we argue that both tasks can be learned and dealt with concurrently, based on the intuition that a word and its definition share the same meaning. We design a neural model to embed words and definitions into a shared semantic space, and generate them from this space. Besides, the training paradigm also includes reconstruction and embedding similarity tasks. Such a system can be viewed as a neural dictionary that supports two-way indexing and searching. Experiments on established datasets demonstrate the ease and effectiveness of our methods. Our model implementation and evaluation scripts are publicly available.[1]

2 Related Work

Although research on the two tasks can be traced back to the early 2000s, recent research has shifted towards neural networks, which we describe here.

Reverse dictionary: Hill et al. (2016) pioneer the use of RNN and bag-of-words models to convert texts to word vectors, on top of which Morinaga and Yamaguchi (2018) add an extra word category classifier. Pilehvar (2019) integrates supersense into target embeddings to disambiguate polysemous words. Zhang et al. (2020) design a multi-channel network to predict a word with its features like category, POS tag, morpheme, sememe, etc.

Nonetheless, our work tackles the problem without disambiguation or linguistics resources. The proposed framework learns autoencodings for definitions and words, instead of mapping texts to plain word vectors. From the aspect of autoencoding, Bosc and Vincent (2018) trains word embeddings via definition reconstruction.

Definition modelling: Noraset et al. (2017) use RNNs for definition generation, followed by Gadetsky et al. (2018) who add attention and word context, and Chang et al. (2018) whose model projects words and contexts to a sparse space and generate from selected dimensions only. Mickus et al. (2019)'s model encodes a context sentence and marks the word of interest, whereas Bevilacqua et al. (2020)'s defines a flexible span of words. Apart from generating definitions freely, Chang and Chen (2019) re-formulated the task to definition retrieval from a closed dictionary, given a word with context.

[1] https://github.com/PinzhenChen/unifiedRevdicDefmod

3 Methodology

3.1 A unified model with multi-task training

In human languages, a word and its definition share the same meaning, despite being represented by different tokens. When approached using a neural method, we hypothesize that a word and its definition can be encoded into a consistent embedding in a semantic space too. This gives rise to our core architecture in the paper: a model that transforms inputs into a shared embedding space that can represent both words and definitions. We then have downstream networks that convert the shared embeddings back to words or definitions. Essentially, a shared representation can be viewed as an autoencoding of the "meaning" of a word and its definition. In the learning process, definition modelling and reverse dictionary are jointly trained to aid each other. At inference time, only half of the network needs to be used.

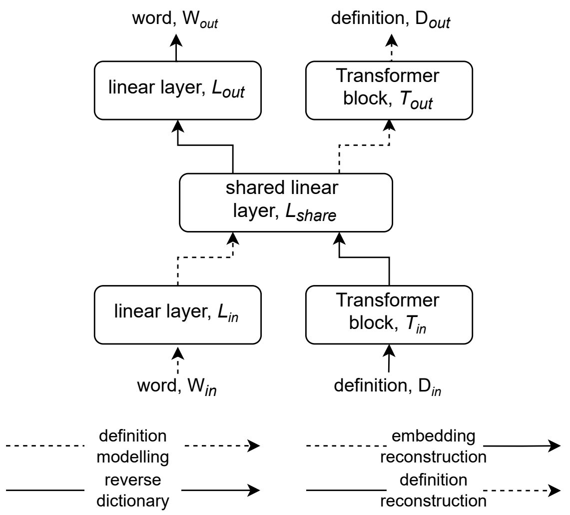


Figure 1: An illustration of our designed model.

The proposed architecture with four sub-task workflows are illustrated in Figure 1. The autoencoding capability is accomplished through a shared linear layer L_{share} between the encoder and the decoder networks, the output of which is the encoded words and definitions. We use linear layers L_{in} and L_{out} to process words W_{in} and W_{out} before and after the shared layer. Likewise, we have definitions D_{in} and D_{out} converted to and from the shared layer, using Transformer blocks T_{in} and T_{out} (Vaswani et al., 2017). In addition, we encourage the shared layer's representations of the input word W_{in} and definition D_{in} to be as close as possible. The Transformer blocks operate on self-attention but not encoder-decoder attention.

With a word embedding distance embed_dist and a token loss token_loss, canonical reverse dictionary and definition modelling will have losses:

$$\mathcal{L}_{\text{revdic}} = \text{embed\_dist}(\mathrm{W}_{gold}, L_{out}(L_{share}(T_{in}(\mathrm{D}_{in}))))$$
$$\mathcal{L}_{\text{defmod}} = \text{token\_loss}(\mathrm{D}_{gold}, T_{out}(L_{share}(L_{in}(\mathrm{W}_{in}))))$$

Our model also optimizes on the losses from word and definition autoencoding:

$$\mathcal{L}_{\text{wordAE}} = \text{embed\_dist}(\mathrm{W}_{gold}, L_{out}(L_{share}(L_{in}(\mathrm{W}_{in}))))$$
$$\mathcal{L}_{\text{defAE}} = \text{token\_loss}(\mathrm{D}_{gold}, T_{out}(L_{share}(T_{in}(\mathrm{D}_{in}))))$$

The distance between the shared word and definition representations is calculated as:

$$\mathcal{L}_{\text{sim}} = \text{embed\_dist}(L_{share}(T_{in}(\mathrm{D}_{in})), L_{share}(L_{in}(\mathrm{W}_{in})))$$

Finally, our training minimizes the overall objective that includes all losses weighted equally:

$$\mathcal{L} = \mathcal{L}_{\text{revdic}} + \mathcal{L}_{\text{defmod}} + \mathcal{L}_{\text{wordAE}} + \mathcal{L}_{\text{defAE}} + \mathcal{L}_{\text{sim}}$$

3.2 Word-sense disambiguation

A word is usually associated with multiple definitions due to the presence of polysemy, sense granularity, etc. In our practice, the one-to-many word-definition relationship does not harm reverse dictionary, where our model can master mapping different definitions into the same word vector. It is problematic for definition modelling, as telling the exact word sense without context is hard. Thus, we embed words in their usage context (provided by the data we use) using BERT (Devlin et al., 2019). Since it operates on sub-words, we sum up the sub-word embeddings for each word.

4 Experiments and Results

4.1 Data and evaluation

HILL: we evaluate reverse dictionary on Hill et al. (2016)'s English data. There are roughly 100k words and 900k word-definition pairs. Three test sets are present to test a system's memorizing and generalizing capabilities: 500 *seen* pairs from training data, 500 *unseen* pairs, and 200 *human description* pairs. The evaluation metrics are ranking accuracies at 1, 10 and 100, as well as the median and standard deviation of the target words' ranks.[2]

CHANG: definition modelling is experimented on Chang and Chen (2019)'s data from the Oxford

[2]Previous papers might use "standard deviation" and "rank variance" interchangeably. We stick to "standard deviation".

	unseen				human description			
	med. rank	acc@ 1/10/100	rank std.†	real std.	med. rank	acc@ 1/10/100	rank std.†	real std.
OneLook.com	-	-	-	-	5.5	.33/.54/.76	332	-
bag-of-words	248	.03/.13/.39	424	-	22	.13/.41/.69	308	-
RNN	171	.03/.15/.42	404	-	17	.14/.40/.73	274	-
category infer	170	.05/.19/.43	420	-	16	.14/.41/.74	306	-
multi-sense	276	.03/.14/.37	426	-	1000	.01/.04/.18	404	-
super-sense	465	.02/.11/.31	454	-	115	.03/.15/.47	396	-
multi-channel	54	.09/.29/.58	**358**	-	**2**	**.32**/.64/.88	203	-
Transformer	79	.01/.14/.59	473	125	27	.05/.23/.87	332	49
unified	**18**	**.13/.39/.81**	386	**93**	4	.22/.64/**.97**	**183**	**30**
+ share embed	20	.08/.36/.77	410	99	4	.23/**.65/.97**	**183**	32

(a) results on HILL with past results from Zhang et al.'s re-run.†They force-set a rank larger than 100 to 1000; we follow suit for comparison, but also include the real std.

	unseen	
	BLEU	ROUGE-L
RNN	1.7	15.8
xSense	2.0	15.9
Transformer	2.4	17.9
unified	2.2	18.5
+ share embed	**3.0**	**20.2**

(b) results on CHANG with past results from Chang and Chen's replicate.

Table 1: Test performance of reverse dictionary (1a on the left) and definition modelling (1b on the right).

English Dictionary. Each instance is a tuple of a word, its usage (context), and a definition. The data has two splits: *seen* and *unseen*. The *seen* split has a training size of 556k, and a test set comprising 9.3k words with 151.3k context and definitions; the *unseen* split consists of 530k training instances, and the test set is 1k words paired with 16.0k context and definitions. The performance is measured by corpus BLEU from NLTK, and ROUGE-L F1[3] (Papineni et al., 2002; Lin, 2004; Bird et al., 2009).

4.2 The questionable *seen* test set

Understandably, a dictionary needs to "memorize" word entries, so both HILL and CHANG supply a *seen* test drawn from the training data. However, this is impractical in deep learning, for it implicitly encourages overfitting. Moreover, the foremost goal of building a neural dictionary is not to memorize seen words; otherwise a traditional rule-based one suffices. Hence, we omit evaluation on the *seen* tests and request future research to not focus on it.

4.3 System configurations

Our baseline is a 4-layer Transformer block: a Transformer encoder for reverse dictionary, and a Transformer decoder for definition modelling. Encoder-decoder attention is not present in any system. Hyperparameters are detailed in Appendix A. We apply a whitespace tokenizer to split all training definitions into an open vocabulary. We use mean squared error (MSE) as the distance function for embeddings, and cross-entropy for definition tokens (the baseline fails to converge with cosine similarity on embeddings).

Our proposed model essentially joins and trains the above two Transformer baselines. The shared layer has the same size as the input embeddings and a residual connection (He et al., 2016). As an additional variant, we tie both Transformer blocks' embedding and output layers (Press and Wolf, 2017). This is only possible with our multi-task framework, since a vanilla Transformer block will have either an encoder or decoder embedding layer but not both. The unified model optimizes roughly twice as many parameters as a single-task baseline; in other words, to perform both tasks, our systems have the same size as the baseline models.

For reverse dictionary, we compare with a number of existing works on HILL: OneLook.com, bag-of-words, RNN (Hill et al., 2016), category inference (Morinaga and Yamaguchi, 2018), multi-sense (Kartsaklis et al., 2018), super-sense (Pilehvar, 2019) and multi-channel (Zhang et al., 2020). Following Zhang et al. (2020)'s treatment, we embed target words with 300d *word2vec* (Mikolov et al., 2013), but definition tokens are encoded from one-hot to 256d, instead of pretrained embeddings. Such an embedding choice ensures a fair comparison to previous works.

For definition modelling on CHANG, words are embedded by 768d *BERT-base-uncased*, while definition tokens are one-hot. We include RNN (Noraset et al., 2017) and xSense (Chang et al., 2018) for reference but not Chang and Chen (2019)'s results from an oracle retrieval experiment.

4.4 Results

Reverse dictionary results in Table 1a show a solid baseline, which our proposed models significantly improve upon. Compared to previous works, we obtain the best ranking and accuracies on *unseen* words. On *human descriptions* our models yield compelling accuracies with the best standard deviation, indicating a consistent performance.

[3] https://github.com/pltrdy/rouge

One highlight is that our model attains a superior position without extra linguistic resources, other than a word embedder which is always used in previous research. Consequently, ours can be a concluded as a more generic method for this task.

Definition modelling results are in Table 1b. On the *unseen* test, our model with tied embeddings achieves state-of-the-art BLEU and ROUGE-L. The model without it has performance similar to the baseline. Nonetheless, the single-digit BLEU hints that the quality of the generation is overall poor.

5 Analysis and Discussions

5.1 Shared embeddings and the vocabulary

For definition modelling, a shared embedding and output layer brings significant improvement to our proposed approach, but in reverse dictionary, our models arrive at desirable results without it. This is reasonable as well-trained embedding and output layers particularly benefit language generation. It further indicates that our multi-task approach is useful, whereby all embedding and output layers share the same weights, in the Transformer sub-models for the two tasks.

We have used an open vocabulary, which has weaknesses like being oversized and vulnerable to unknown tokens. Therefore, we add a model with a 25k unigram SentencePiece vocabulary (Kudo and Richardson, 2018) to definition modelling. All other configurations remain the same as the best-performing model. BLEU and ROUGE-L drop to 2.5 and 18.7, implying that an open vocabulary is not an issue in our earlier experiments.

5.2 Human evaluation on definitions

For definition modelling, we notice that the low BLEU may not be indicative. As a further investigation, we conduct both reference-less and reference-based human evaluation, on the Transformer baseline and the best-performing unified model. In a *reference-less* evaluation, a human sees a source word, and picks the preferred definition output, whereas in a *reference-based* setting, a human sees the reference definition instead. In each setting, test instances are sampled, then the models' outputs are presented in a shuffled order to evaluators. Two annotators, in total, evaluated 80 test instances for each setting. We record the number of times each model is favoured over the other in Table 2.

Regardless of the evaluation settings, the human evaluators favour our model's outputs over the base-

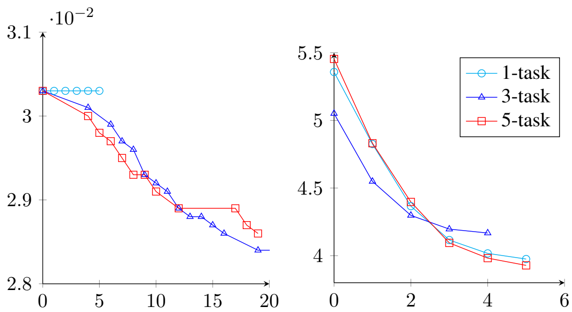
	reference-less	reference-based
Transformer	25 (31%)	32 (40%)
unified + share embed	**50 (63%)**	**42 (53%)**

Table 2: Chances a model's output is preferred by human evaluators. Columns do not add up to 80 (100%), because we do not count cases where both models output the same.

line's. Specifically in the reference-less evaluation, which resembles a real-life application of definition generation, our proposed model wins notably.

5.3 Ablation studies on the objectives

Our models are trained with five objectives from five tasks: definition modelling and reverse dictionary, two reconstruction tasks and a shared embedding similarity task. We design an ablation study to understand how multi-task learning contributes to performance. We designate our unified model on HILL's reverse dictionary with shared embeddings a "5-task" model. From there, we exclude word and embedding reconstruction by disabling respective losses to form a "3-task" model. Further, we build a single-task model by removing definition modelling and embedding similarity losses. We then run similar experiments for definition modelling. We plot the statistics during training in Figure 2: embedding MSE against epochs for reverse dictionary, and generation cross-entropy against epochs for definition modelling. The curve plotting stops when validation does not improve.



(a) reverse dictionary (b) definition modelling

Figure 2: Validation losses (y-axis) against epochs (x-axis).

As Figure 2a shows, the single-task HILL model does not converge, probably because in reverse dictionary the Transformer block is far away from the output end, and only receives small gradients from just one loss. The 3-task and 5-task models display similar losses, but the 3-task loss curve is smoother. In Figure 2b for definition modelling, the 3-task model trains the fastest, but 1-task and 5-task models reach better convergence. The analysis implies that training on two tasks is always beneficial, and reconstruction is helpful but not crucial.

6 Conclusion

We build a multi-task model for reverse dictionary and definition modelling. The approach records good numbers on public datasets. Also, our method delegates disambiguation to BERT and minimizes dependency on linguistic resources, so it can potentially be made cross-lingual and multilingual. A limitation is that the evaluation centres on English, without exploring low-resource languages.

Acknowledgements

Pinzhen Chen is supported by funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825303 (Bergamot). Zheng Zhao is supported by the UKRI Centre for Doctoral Training in Natural Language Processing (UKRI grant EP/S022481/1). This work reflects the view of the authors and not necessarily that of the funders.

References

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or "how we went beyond word sense inventories and learned to gloss". In *Proceedings of EMNLP*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*.

Tom Bosc and Pascal Vincent. 2018. Auto-encoding dictionary definitions into consistent word embeddings. In *Proceedings of EMNLP*.

Ting-Yun Chang and Yun-Nung Chen. 2019. What does this word mean? explaining contextualized embeddings with natural language definition. In *Proceedings of EMNLP-IJCNLP*.

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xSense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *arXiv*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of ACL*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *TACL*, 4:17–30.

Dimitri Kartsaklis, Mohammad Taher Pilehvar, and Nigel Collier. 2018. Mapping text to knowledge graph entities using multi-sense LSTMs. In *Proceedings of EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of EMNLP*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*.

Timothee Mickus, Denis Paperno, and Matthieu Constant. 2019. Mark my word: A sequence-to-sequence approach to definition modeling. In *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR Workshop*.

Yuya Morinaga and Kazunori Yamaguchi. 2018. Improvement of reverse dictionary by tuning word vectors and category inference. In *International Conference on Information and Software Technologies*.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of AAAI*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*.

Mohammad Taher Pilehvar. 2019. On the importance of distinguishing word meaning representations: A case study on reverse dictionary mapping. In *Proceedings of NAACL*.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of EACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Hang Yan, Xiaonan Li, Xipeng Qiu, and Bocao Deng. 2020. BERT for monolingual and cross-lingual reverse dictionary. In *Findings of EMNLP*.

Lei Zhang, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2020. Multi-channel reverse dictionary model. In *Proceedings of AAAI*.

A Hyperparameters and Computation

Our model configuration and the tuning process is summarized here. We adjusted the hyperparameters on the baseline model using the validation set, and kept the values unchanged for the proposed model which contains two baseline Transformer blocks. We list the hyperparameters in Table 3, and highlight the selected ones in bold if multiple runs/values are tried out. Instead of an expensive grid-search on all combinations, we searched for the best configurations one by one.

A HILL model has 35.1M parameters in total, and a CHANG model has 62.7M. On a single Nvidia GeForce GTX 1080 Ti, a HILL experiment takes about 1 hour/epoch and on average converges after 60 epochs. All CHANG models are trained on a single Nvidia GeForce RTX 2080 Ti at 1 epoch/hour, and a model needs roughly 6 epochs to converge. We only performed a single run for each experiment.

word embed.	HILL: word2vec CHANG: BERT-base-uncased
word embed. dim.	HILL: 300 CHANG: 768
definition tokenizer	whitespace
def. token embed.	none, trained from one-hot
def. token embed. dim.	256
training toolkit	PyTorch (Paszke et al., 2019)
stopping criterion	5 non-improving validations
learning rate	1e-3, **1e-4**, 1e-5 and 1e-6
optimizer	Adam (Kingma and Ba, 2015)
beta1 and beta2	0.9 and 0.999
weight decay	1e-6
embedding loss	**MSE**, cosine
token loss	cross-entropy
training batch size	HILL: 256 CHANG: 128
decoding batch size	1
decoding beam size	**6**, 64
Transformer depth	**4**, 6
Transformer head	**4**, 8
Transformer dropout	0.1, **0.3**
def. token dropout	**0**, 0.1
linear layer dropout	0.2
linear layer dim.	HILL: 256 CHANG: 768
shared layer dim.	HILL: 256 CHANG: 768
trainable parameters	HILL: 35.1M CHANG: 62.7M

Table 3: Model and training configurations.